[illegible]

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/898,351	07/03/2001	Pratap Subrahmanyam	10991880	1056

7590 01/13/2005

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, CO 80527-2400

EXAMINER	
VU, TUAN A	
ART UNIT	PAPER NUMBER
2124	

DATE MAILED: 01/13/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/898,351	SUBRAHMANYAM ET AL.	
	Examiner	Art Unit	
	Tuan A Vu	2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 28 September 2004.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1,2,4-12,14-17,19 and 20 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1,2,4-12,14-17,19 and 20 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 9/28/2004.

As indicated in Applicant's response, claims 1-2, 4-5,8-12, 14-17, 19-20 have been amended and claims 3, 13, 18 canceled. Claims 1-2, 4-12, 14-17, 19-20 are pending in the office action.

Claim Rejections - 35 USC § 112

2. The following is a quotation of the first paragraph of 35 U.S.C. 112:

The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

3. Claims 7-9 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention.

The elements recited as

'...said counter cache is full' (claims 7-9, li. 2, li. 4, li. 7, respectively) are not supported by any explicit description in the specification.

The specification makes it clear in Fig. 4 via pg. 11-12, and Fig. 8 via pg. 15-16, that it is code cache 27(Fig. 3-4) that is determined to be full for code eviction purposes. Code cache (*code cache 27*) and *counter cache 25* as exhibited in Fig. 1 and 3, are two different entities and there is no description about checking if a so-called 'counter cache' is full.

A non-compliance defect of a same type/nature was brought to the Applicants' attention as a rejection in the previous Office Action. It is the responsibility of the Applicants to ensure that all the claims are cleared from a particular impropriety/violation type being rejected. Since Applicants' amendment seem to have overlooked claims 7-9, the above non-compliance defects are again rejected for the same reasons as in the last action.

The interpretation will be at best based on the understanding derived from the specification, i.e. one alternative way of interpreting being to equate *counter cache* to *code cache*, even though other ways of interpreting are possible.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1-2, 4-5, 11-12, 14-17, and 19-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Tremblay et al., USPN: 6,065,108 (hereinafter Tremblay), in view of Burton et al., USPubN: 6,738,865 (hereinafter Burton).

As per claim 1, Tremblay discloses a method to analyze a computer program that includes a plurality of block of code, the method comprising :

receiving a code instruction to a code cache (e.g. cache 125, 134, 153, 155 – Fig. 1; Fig. 3-4c; Fig. 6);

using a counter for tracking each time said code instruction is loaded on said code cache (e.g. *cache 153 is loaded ... is a counter* - col. 20, lines 14-28 – Note: loading a instruction and determine whether its arguments are ready using a counter reads on tracking each time a code instruction is executed);

maintaining a counter cache (e.g. col. 20, lines 14-28) for storing each said counter of said code instruction while said code instruction is stored on said code cache.

But Tremblay does not explicitly disclose that the instructions cache is for storing block of code. But if code is only a machine representation of some human-readable sequence of programming language, then programming instructions can be a block of such code as in basic block; therefore Tremblay has disclosed block of code in code cache.

Nor does Tremblay explicitly disclose tracking each time that said block of code is executed to maintain a count therefor. Tracking code execution frequency to allow the use of code cache replacement was a known concept; and Tremblay does teach such replacement policy (col. 28, line 63 to col. 28, line 9) while Burton, using also a LRU policy teaches a counter to tracking block of code execution (see *counter 18* – Fig. 1). Based on known reasons why cache replacement are so important, in case Tremblay does not already provide a count for the number of executed code being cached, it would have been obvious for one of ordinary skill in the art at the time the invention was made to provide such tracking of execution frequency as taught by Burton, because with this tracking, the LRU policy intended by Tremblay would be able to operate, since it is the frequency of the most or least recently executed instructions that direct how the cache replacement of Tremblay would be successful.

Nor does Tremblay disclose maintaining a storage area for storing each said counter of said block of code previously executed on said code cache after said block of code is evicted from said code cache. The use of eviction of least used data in fast memory was a well-known concept via the LRU algorithm methodology to manage cache resources such as evidenced by Tremblay (e.g. col. 28, line 63 to col. 28, line 9), such removing code from cache entailing a storage area to keep the code thus demoted from the LRU policy, and the use of look up table or associative fast memory as shown by Tremblay is suggestive of similar storage of instructions or data that are most often executed or less frequently so (e.g. col. 20, line 39 to col. 21, line 20; col. 30, lines 18-25). Further, in a system to correspond executed code in code cache with a counter analogous to Tremblay, Burton also discloses a similar storage of table reminiscent to the look up table by Tremblay, including ordered counter to track cache entries implemented by a storage/list showing entries with their LRU ranking; and demoting those entries from high ranking from cache (in case cache is low in memory) by storing them in lower of such list, i.e. demoted entries amounting to a count record (Fig. 1, *LRU linked list*; Fig. 3). This is similar to maintaining a count of instructions being evicted from cache high LRU policy ranking because the demoting implicitly discloses removing instructions/data from cache (see Fig. 2-3). It would have been obvious for one of ordinary skill in the art at the time the invention was made to ensure that the instructions being evicted from cache following Tremblay LRU policy and look up table techniques also implement the counting record of those cache entries that have been evicted or demoted as taught by Burton because the recorded evicted entries being tagged with some status number reflecting their previous execution count/frequency can inform on how the cache should keep entries without having to continually allocate resources to keep track of the dynamic execution frequency of instructions(see Burton BACKGROUND, and Summary).

As per claim 2, based on the known concept to apply LRU policy to prevent cache overstorage, the teachings by Tremblay regarding LRU (see claim 1) implies the desirability to keep the code cache from being full. Further, Burton disclose such concept of determining whether a cache is full (see Fig. 3). It would have been obvious for one of ordinary skill in the art at the time the invention was made to provide Tremblay LRU's policy the additional step of the cache threshod determining as taught by Burton, because providing a replacement policy to alleviate burden of a cache requires means of finding when cache resources reach unwanted threshold as shown by Burton or implied by well-known methodology to apply LRU replacement from above, because there would be no need to apply a replacement algorithm when no resources are threatened.

As per claim 4, the combination of Tremblay/Burton discloses determining which of said counter of said block of code stored on said counter cache is least recently executed; and evicting said least recently executed block of code related to said counter from said code cache (re claim 1 and the LRU related teachings by Tremblay and Burton). Further, the rationale as to combine the record of demoted entries by Burton in conjunction with the lookup table or associative cache by Tremblay has rendered obvious the limitation as to copying said counter of said least recently executed block of code from said counter cache to said storage area when said least recently executed block of code related to said counter is evicted from said code cache (re corresponding rejection in claim 1).

As per claim 5, Tremblay discloses

checking said storage area to determine if said block of code is being executed for other than the first time (e.g. col. 20, lines 14-28 – Note: the look aside for variables/arguments for a given method being loaded in the instruction cache and tracking the count of such look up for the first time when said method is assembled reads on checking said storage area to determine if said block of code is being executed for other than the first time; loading said counter associated with said block of code being executed for other than the first time – see Fig. 2-4D);

But Tremblay does not explicitly disclose checking said storage area to determine if said block of code is being executed for other than the first time; loading said counter associated with said block of code being executed for other than the first time; and updating said counter associated with said block of code being executed for other than the first time. In combination with finding out how many times a instruction (see *counter 18* – Fig. 1; Fig. 2-3) has been executed for tracking its status in regard to a LRU replacement policy as taught above by Burton, the checking to see if a block of code is executed more often than a first time is disclosed. Hence, this limitation is implicitly disclosed or would have been obvious in view of the demoting and promoting by Burton (re claim 1) and in view of Tremblay's LRU policy in conjunction with reading a look up table or associative cache (col. 20, line 39 to col. 21, line 20; col. 30, lines 18-25).

As per claim 11, this is a computer readable medium having computer readable program code embodied therein to perform when executed a method for analyzing a computer program that includes a plurality of blocks of code comprising logic to perform the same steps as recited in claim 1; hence is rejected with the corresponding rejection as set forth therein.

As per claims 12, 14, and 15, these claims correspond to claims 2, 4, and 5, respectively; hence are rejected using the corresponding rejections as set forth therein.

As per claim 16, this claim is the system version of claim 1, hence is rejected with the corresponding rejection as set forth therein.

As per claims 17, 19, and 20, these claims correspond to claims 2, 4, and 5, respectively; hence are rejected using the corresponding rejections as set forth therein

6. Claims 6-10 are rejected under 35 U.S.C. 103(a) as being unpatentable over Holmberg et al., USPubN: 2001/0021959 (hereinafter Holmberg), in view of Burton et al., USPN: 6,738,865.

As per claim 6, Holmberg discloses a method to analyze a computer program that includes a plurality of blocks of code (pg. 3, para 0028), the method comprising means for:

executing said computer program (e.g. pg. 3 para 0028, pg. 5, para 0050);

using a counter for tracking each time one of said plurality of blocks is executed (e.g. *access counter 25* -pg. 3 para 0034; *single basic block* - pg. 2, para 0013);

maintaining a counter associated with a cache for storing said plurality of count references in a cache of said blocks of code that are most recently executed (e.g. *reference number 23, counter 25* – Fig. 1a).

But Holmberg does not disclose maintaining a counter cache for storing the counter of the plurality of blocks of code that are most recently executed. Holmberg teaches placing in cache frequently used data (e.g. pg. 5, para 0054), linking access allocation with table and suggests storing *link table* in cache (pg. 3, para 0035), hence has disclosed storing in cache information related to corresponding counter of blocks that are most recently executed. Further, organizing cache for code for execution so that only most executed code is cached and that the least recently used cache entries are replaced using some algorithm, e.g. LRU replacement, was known concepts by the time the invention was made. Burton, in a method to maintain cache for execution involving memory access operations using counter analogous to Holmberg's approach, discloses a counter placed in cache for effecting listing of cache data according to the most frequently used priority manner, i.e. a counter cache to associate cache entries being ordered by the most recently accessed rank (e.g. Fig. 1-2). In view of well-known techniques of keeping in cache information that would directly support cache replacement policy such as cache look-up tables, associative cache arrays or particularly linked list array as by Burton (see LRU list 10 – Fig. 1), it would have been obvious for one of ordinary skill in the art at the time the invention was made to modify the counter means by Holmberg to implement the counter by Burton so that the cache now stores the counter for tracking the frequency of data access or execution, hence would expedite the process of replacing least frequently used versus most frequently used data in the cache by the support of cache storage of important data teaching on priority of data to be retained in cache as taught by Burton, especially to enhance the desirability of caching such data as addressed by the static cache by Holmberg (see Holmberg: col. 2, para 0015) and also in view of well known practices as mentioned above.

Nor does Holmberg disclose maintaining a storage area for storing counters associated with blocks of code that are not most recently executed. Burton discloses a storage area to store entries that have been demoted from the replacement in the cache based on the counter association with the priority given to most frequently accessed entries (e.g. *storage 6* - Fig. 3; col. 4, lines 6-34), while Holmberg implements 2 memories to support each other in optimizing cache loading with support of counters (MM, SC, Fig. 1a-b) and multi-level cache (pg. 7, para 0080). The concept of moving data not appropriate for a fast cache to a lower level memory is hence strongly evident. It would have been obvious for one of ordinary skill in the art at the time the invention was made to use Burton's approach of maintaining the demoted blocks in a storage area and add this to the maintaining of counter in the first cache as set forth above by

Holmberg/Burton so that counters in data in the first cache (those cache blocks/entries that are not recently executed) can be demoted to be stored in the second memory as suggested by Holmberg, accomplishing thereby dynamically the cache replacement scheme while preserving the least recently accessed blocks for future reference, i.e. a concept known in the art of cache temporal and spatial replacement.

As per claim 7, official notice is taken that the replacement technique, e.g. LRU replacement algorithm, applied to cache when it is determined that cache capacity is nearly exhausted was a well known concept at the time the invention was made. Even though Holmberg does not explicitly disclose determining when a counter cache is full, in view of the teachings by Holmberg and Burton to manage the cache, it would have been obvious for one of ordinary skill in the art at the time the invention was made to add the limitation of determining when the cache is full as well known in the art and implement that to Holmberg/Burton's combination so to help anticipating resources exhaustion and rectify the contents of the cache in a timely fashion to avert memory crash when in light of well-known methodologies as to apply LRU replacement from above, there would be no need to apply a replacement algorithm when no resources are threatened.

As per claim 8, official notice is taken that in the LRU replacement policy, the moving of data deemed not high in priority from cache to another memory was a known concept in any cache replacement practice. Hence, in view of the techniques taught by Burton (see pg. 7, para 0080) and the rationale in claim 7, the limitation of copying blocks from counter cache to a storage area when the counter cache is full would also be if not implicitly disclosed by Holmberg; then obvious in light of the knowledge from the above notice and Burton' s teachings (see Burton: *storage 6* - Fig. 3; col. 4, lines 6-34). One skill in the art would apply the cache policy as suggested via Holmberg's technique to keep the most used data in cache, i.e. store away the least used data to a storage when the cache threshold is threatened as taught by well-known practices or the demoting for LRU as by Burton and via the moving of less needed data to another level of memory as in the multi-level cache by Holmberg, because temporally not used data can be stored away from fast cache to unburden cache overload so to make room for most frequently used data therein; obviating cache limit violation while the stored away data can be reused when a cache hit or miss would necessitate their re-caching, as suggested by Holmberg multi-cache method or Burton's LRU cache-and-memory transaction according to the well-known concepts mentioned above.

As per claim 9, Holmberg teaches maintaining in cache the most frequently executed data and keep less used data in a slower memory (pg. 5, para 0050; pg. 7, para 0080) and Burton teaches cache recording of LRU(least recently used) information (Fig. 2) in conjunction with demoting cache entries so they are stored into secondary area (re claim 1). The limitation as to determining which of the counting means of code blocks is least recently used and copying said block of code to a storage area when the cache is full would also have been either implicitly disclosed by Holmberg or obvious using the rationale as set forth in claim 8 above.

As per claim 10, Holmberg does not explicitly teach checking counter cache to determine if a block is being executed other than a first time; and loading a counter associated therewith into the counter cache; but the concept to keep cache based on spatial and temporal (checking if a data is being used more than one time) proximity was a known concept at the time the invention was made. Holmberg however teaches an access counter (pg. 3, para 0034).

Hence, the checking to determine if a block of code is executed other than a first time is disclosed.

The teaching of counter cache (i.e. loading said counting means associated with said block of code executed for other than a first time, into said counter cache) has been addressed in claim 6 using Burton's LRU linked list. Hence, it would have been obvious for one of ordinary skill in the art at the time the invention was made to include in Holmberg's method the counter cache as taught by Burton using the rationale as already set forth in claim 6 because a counter with a time tracking as taught by Burton would enhance the determination in the least temporally accessed (i.e. code being executed for other than the first time) data and that cache information would speedily enable which data to be kept or evicted for the same reasons as set forth in claim 6 in light of the desirability for caching important data as addressed by the static cache by Holmberg (see Holmberg: col. 2, para 0015).

Response to Arguments

7. Applicant's arguments with respect to claims 1-20 have been considered but for the most part are moot in view of the new ground(s) of rejection.

As for Applicant's arguments on Holmberg's teaching away by not relying/using solely of frequency to manage code cache (Appl. Rmrks, pg. 13, 3rd para), it is noted that the concept of keeping in cache a counter is at stakes here. The argument concerning Holmberg's disregarding the importance of frequency of code access is not persuasive. Holmberg discloses a counter to keep track of access of code or variables, thus used for advice to allocate memory 12 as Fig. 1 (see pg. 3 para 0034); and this put forth access frequency of code blocks. Holmberg further associate table to link such code access tracking with memory allocation via cache storing of such link information; hence has suggested a cache of data related to the number of access by some block of code. Therefore, using such suggestion, the rationale as to render obvious the limitation referred to 'counter cache' has been set forth using a motivation based on Holmberg's suggestion combined with known concept and Burton's teachings. Applicants hence fail to point out why such motivation for combining as set forth in the rejection would generate unwanted results or adverse teachings to the endeavors of Burton or Holmbergs.

The rejection therefore will stand as set forth.

Conclusion

8. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (272) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (571)272-3719.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence – please consult Examiner before using) or 703-872-9306 (for official correspondence) or redirected to customer service at 571-272-3609.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

VAT
December 31, 2004

[illegible]

TODD INGBERG
PRIMARY EXAMINER